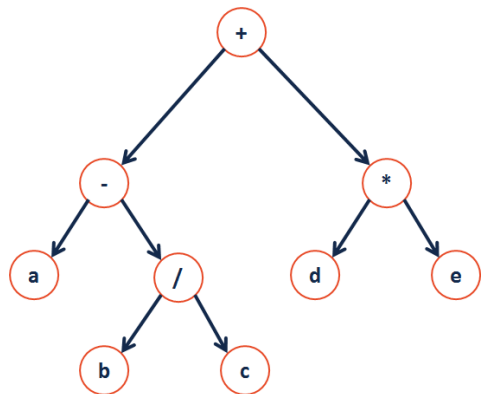


Traversals:

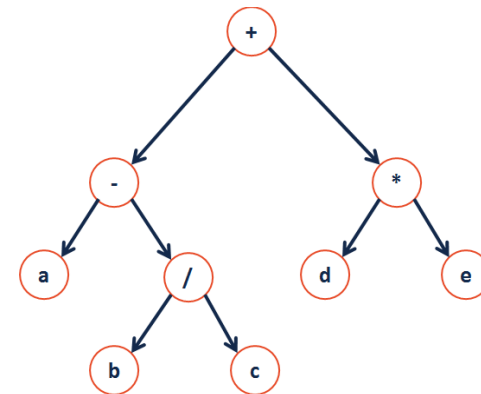


One Algorithm, Three Traversals:

BinaryTree.cpp	
50	void BinaryTree<T>::Order(TreeNode * cur) {
51	if (cur != nullptr) {
52	
53	
54	
55	
56	
57	}
58	}

A Different Type of Traversal

Strategy:



BinaryTree.cpp	
	void BinaryTree<T>::levelOrder(TreeNode * croot) {
	}

Traversal vs. Search:

- **Traversal** visits every node in the tree exactly once.
- **Search** finds one (or more) element(s) in the tree.

Breadth First Traversal + Search:

Depth First Traversal + Search

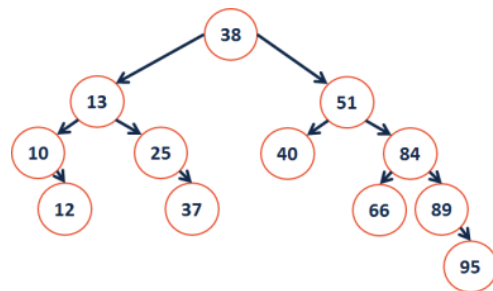
Runtime Analysis on a Binary Tree:

- Find an element: Best case? Worst case?
- Insertion of a sorted list of elements?
Best case? Worst case?
- Running time bound by

Dictionary ADT

```
Dictionary.h
3
4 class Dictionary {
5     public:
6
7
8
9
10
11
12
13     private:
14
15
16 };
```

A Searchable Binary Tree?



Binary Search Tree Property:

Finding an element in a BST:

```
BST.hpp
template <typename K, typename V>
_____ find(const K & key) {
}
template <typename K, typename V>
_____ _find
(TreeNode *& root, const K & key) {
}
}
```

CS 225 – Things To Be Doing:

1. mp_list due Sunday.
2. lab_inheritance starts today
3. exam 1 reschedule window Saturday 2/26 – Monday 2/28.
4. Daily POTDs